Knowledge Base Repair: From Active Integrity Constraints to Active TBoxes

Guillaume Feuillade[1], Andreas Herzig[1], and Christos Rantsoudis[2]

[1] IRIT, CNRS, Univ. Toulouse, 31062 Toulouse Cedex 9, France,
firstname.surname@irit.fr
[2] School of Computing Science, Simon Fraser Univ., Burnaby, BC, V5A 1S6, Canada,
firstname_surname@sfu.ca

Abstract. In the database literature it has been proposed to resort to active integrity constraints in order to restore database integrity. Such active integrity constraints consist of a classical integrity constraint together with a set of preferred update actions that can be triggered when the constraint is violated. In this overview paper we start by reviewing the main repairing routes that have been proposed in the literature. We do so from the perspective of Dynamic Logic, viewing active integrity constraints as programs that test whether a constraint is violated and if so perform appropriate update actions. We then discuss how these ideas can be adapted to Description Logics. We assume extensions of TBox axioms by update actions that denote the preferred ways an ABox should be repaired in case of inconsistency with the axioms of the TBox. The extension of the TBox axioms with these update actions constitute new, active TBoxes.

Keywords: database repair, knowledge base repair, active integrity constraints, description logic, dynamic logic

1 Introduction

One of the most important and notoriously difficult issues in the database and AI literature is the problem of updating a database under a set of *integrity constraints*. In the course of database maintenance several changes are applied to the databases and checking whether these constraints are still entailed is of fundamental importance.

The TBoxes of Description Logic knowledge bases bear resemblances with integrity constraints. However, due to the open world assumption the TBox axioms do not have to be entailed by the ABox; instead, their role is to infer more than what is literally written in the ABox. TBoxes therefore only have to be consistent with the ABox. Hence the counterpart of checking whether the

updated database still entails the integrity constraints is the problem of checking whether the updated ABox is still consistent with the TBox.[1]

When a database fails to satisfy the integrity constraints then there are basically two different options: one may either live with the inconsistency or repair the database in order to restore integrity. If one chooses the latter option then one has to explicitly construct the repaired database. The former option requires the definition of non-standard inconsistency-tolerant semantics that are *a priori* in the spirit of paraconsistent logics; one may in particular rely on hypothetical repairs. The two options therefore lead to solutions that are technically quite similar.

Procedures restoring the consistency of a database with respect to a set of integrity constraints were extensively studied in the last decades [1, 9, 3]. These approaches propose several possible repairs as candidates for integrity maintenance and it seems essential to identify which types of repairs are more suitable, all the more the number of all possible repairs can be remarkably large. Given this, the most prevalent have become those that are based on the *minimality of change* principle [30, 17, 10].

The situation is similar for description logic knowledge bases when an ABox is updated in a way such that it becomes inconsistent with a given TBox. For DL knowledge bases, the most widespread methods to repair inconsistencies are based on the so-called *justifications*, which are minimal subsets of the KB containing the terminological and assertional axioms from which an undesirable consequence is inferred. Axiom pinpointing through justifications became an important topic of research within the DL community and several results were quickly established [25, 18, 27]. Both black-box [26, 2] and glass-box [21, 19] methods emerged for computing justifications. The former have a more universal approach and are used independently of the reasoner at hand, while the latter have a more delicate construction that is tied to specific reasoners and usually require less calls. After computing all justifications of an undesirable consequence, the next step is to obtain a minimal *hitting set* [24] made up of one axiom per justification and remove it from the knowledge base. More recent approaches have focused on providing methods for *weakening* the axioms instead of removing them, since the latter can prove to be too big of a change [29]. Another research avenue investigates repair-based semantics for query answering [4, 5].

It seems fair to say that for the time being there is no consensus on the 'right' way of repairing inconsistent knowledge bases. Given these long-standing difficulties to define appropriate repairs, it seems reasonable to resort to more modest approaches where more information is fed into the repair process. In the present paper we take inspiration from a research avenue in the database literature where integrity constraints are equipped with additional information about the preferred way to maintain consistency: in [16] *Active Integrity Constraints*

[1] We note that there are approaches combining integrity constraints with TBoxes. The most prominent ones are based on the extension of KBs with *constraint axioms*. These axioms are however used for validation purposes only and do not share the same semantics as regular TBox axioms [20, 28].

($\mathcal{AIC}$s) were proposed as an enhancement of classical integrity constraints with update actions. This research avenue was further pursued in a series of publications [7, 8, 11]. $\mathcal{AIC}$s extend classical integrity constraints with *preferred* update actions. This introduces a dynamic view of the concept of integrity constraint. From now on we therefore call classical integrity constraints *static* constraints and will talk about the *static part* of an $\mathcal{AIC}$.

For instance, consider the classical integrity constraint:

$$\forall(E, S_1, S_2)[Employee(E, S_1) \wedge Employee(E, S_2) \rightarrow S_1 = S_2]$$

which says that every employee should only have one salary. It can be extended into the active constraint:

$$\forall(E, S_1, S_2)[Employee(E, S_1) \wedge Employee(E, S_2) \wedge S_1 > S_2 \rightarrow -Employee(E, S_1)]$$

which states that if there is an employee with two salaries then the preference is to remove the highest salary (instead of removing one randomly).

Another example is the static integrity constraint:

$$(\forall X)[\mathsf{Bachelor}(X) \wedge \mathsf{Married}(X) \rightarrow \bot]$$

which says that no one should have the property of being a bachelor and married at the same time. It can be turned into the active constraint:

$$(\forall X)[\mathsf{Bachelor}(X) \wedge \mathsf{Married}(X) \rightarrow \bot, \{-\mathsf{Bachelor}(X)\}]$$

whose meaning is that when there is a person in the database who has both the status of being a bachelor and the status of being married then the preferred repair is to remove from the database the bachelor status (as opposed to removing the married status) since married status can be achieved from being bachelor but not the other way. In this way, the possible repairs better match designer preferences when maintaining the database. Moreover, it can be expected that their number is narrowed down.

In the propositional case, an active integrity constraint can be represented as a couple:

$$r = \langle \mathsf{C}(r), \mathsf{R}(r) \rangle$$

where $\mathsf{C}(r)$ is a boolean formula (called the static part of r and denoting a static constraint) and $\mathsf{R}(r)$ is a set of update actions, each of which is of the form $+p$ or $-p$ for some atomic formula p. The idea is that (1) when the static constraint $\mathsf{C}(r)$ is false then the constraint r is violated, and (2) a violation of r can be repaired by performing one or more of the update actions in $\mathsf{R}(r)$.

2 The Semantics of Active Integrity Constraints

It remains to give a semantics to a given set of active integrity constraints in terms of the repairs it induces. The two most important ones that were proposed in the literature are the *founded* and the *justified* repairs. While both semantics

greatly reduce the number of possible repairs, different choices between update actions in $\mathsf{R}(r)$ can still lead to different repairing routes, or even no repairs at all. This is for example trivially the case when $\mathsf{R}(r)$ is the empty set.

In the first paper about $\mathcal{AIC}$s the authors introduce *founded* repairs [16]. The idea is that any update action applied to the database should be *supported* by the 'active part' of an $\mathcal{AIC}$, i.e., by a preferred update action of a violated constraint. They also obtain complexity results for the problem of existence of founded repairs, both in the general case (Σ_P^2-complete) as well as in the case that $\mathcal{AIC}$s comprise 'single heads', i.e., only one preferred update action is allowed in each constraint (NP-complete). Recognizing that the existence of founded repairs is not always guaranteed though, they go on to define *preferred* repairs as an intermediate repairing route between founded and standard repairs that always exist. The complexity for the problem of existence of preferred repairs is shown to be Σ_P^2-complete. Further research on $\mathcal{AIC}$s also ensued [6, 7] that reviewed and expanded upon the aforementioned results.

The first new definitions on repairing procedures that are based on preferences between update routes came in [8], an attempt to relate the seemingly different approaches to $\mathcal{AIC}$s and *revision programming*. There, the authors distinguish between the various repairs that they propose (standard repairs, founded repairs, justified repairs) and their *weaker* versions (weak repairs, founded weak repairs, justified weak repairs), with only the former complying to the *minimality of change* principle that the previous approaches took by default. The definitions of justified weak repairs and justified repairs were introduced as a response to the so-called *circularity of support* defect that founded repairs cannot evade and which the authors argue against. Furthermore, they leave the first-order setting of the previous papers and use a propositional one. The propositional setting of [8] provides a valuable stepping stone to present and discuss our own approach. Another distinguishing feature in their work is that they investigate properties of *normalization*, i.e., 'breaking' all active integrity constraints into many copies such that each one has at most one preferred update action. They denote the differences that exist in these two different classes of $\mathcal{AIC}$s, both in practice (resulting in more or fewer repairs) as well as in the complexity of deciding the existence of repairs under *normal* $\mathcal{AIC}$s. The consensus on complexity results is very interesting, with all of the different kinds of repairs, being either weak or minimal, and being applied on either normal $\mathcal{AIC}$s or standard $\mathcal{AIC}$s, to fall either on the NP-complete or Σ_P^2-complete territory.

Last but not least, further approaches to refine or extend active integrity constraints have been investigated in [12, 11] with analyses of algorithms on trees, extensions to knowledge bases outside databases, as well as independence/precedence relations among active integrity constraints.

In the paper [15] we have recast these different semantics in Dynamic Logic. More precisely, we have shown that active integrity constraints can be viewed as particular programs in Dynamic Logic of Propositional Assignments DL-PA: they consist in a test of the negation of the static part of the $\mathcal{AIC}$ that is followed by a nondeterministic composition of the possible update actions. The latter are

identified with sets of assignments of propositional variables to truth values. Given an $\mathcal{ALC}$ $r = \langle \mathsf{C}(r), \mathsf{R}(r) \rangle$, the associated DL-PA program is therefore:

$$\neg \mathsf{C}(r)\,?\,;\bigcup \mathsf{R}(r)$$

These are the building blocks of programs capturing all the above semantics, as shown in [15]. Furthermore, a new kind of repair program is defined there as:

$$\bigcup_{r}\Big(\neg \mathsf{C}(r)\,?\,;\bigcup \mathsf{R}(r)\Big)$$

and is compared to the existing semantics.

3 From Active Integrity Constraints to Active TBoxes

It seems natural to us to transfer the active integrity constraint idea to the Description Logic setting. If we identify ABoxes with databases and TBoxes with integrity constraints then the axioms of the latter should be extended with preferred update actions. These update actions should prescribe the changes the ABox should undergo in order to reestablish consistency whenever it is inconsistent with the TBox. We call these extended TBoxes *active TBoxes*. They enable ontology engineers to easily express preferred update actions, thereby avoiding that unwanted repairing routes are followed. The idea is to repair inconsistent KBs by updating ABox assertions in order to comply with (preferred update actions indicated by) the axioms of the active TBox. Somewhat surprisingly, this research avenue had not been tried in the DL literature.

For example, consider the following static TBox:

$$\mathcal{T} = \{\mathsf{Father} \sqsubseteq \mathsf{Male} \sqcap \mathsf{Parent}, \mathsf{OnlyChild} \sqsubseteq \forall \mathsf{hasSibling}.\bot\}$$

An ontology engineer may wish to enhance it to two possible *active* TBoxes, viz. $\mathsf{a}\mathcal{T}_1 = \{\eta_1, \eta_2\}$ and $\mathsf{a}\mathcal{T}_2 = \{\eta_3, \eta_4\}$ where:

$$\begin{aligned}
&\eta_1 : \langle \mathsf{Father} \sqsubseteq \mathsf{Male} \sqcap \mathsf{Parent}, \{+\mathsf{Male}, +\mathsf{Parent}\}\rangle;\\
&\eta_2 : \langle \mathsf{OnlyChild} \sqsubseteq \forall \mathsf{hasSibling}.\bot, \{-\mathsf{OnlyChild}\}\rangle;\\
&\eta_3 : \langle \mathsf{Father} \sqsubseteq \mathsf{Male} \sqcap \mathsf{Parent}, \{-\mathsf{Father}\}\rangle;\\
&\eta_4 : \langle \mathsf{OnlyChild} \sqsubseteq \forall \mathsf{hasSibling}.\bot, \{-\mathsf{hasSibling}.\top\}\rangle
\end{aligned}$$

We can witness how, through these enhanced concept inclusions, one can be more specific in the update actions that s/he prefers when repairing an ABox that is inconsistent with $\mathcal{T}$: the active axioms of $\mathsf{a}\mathcal{T}_1$ dictate that an individual who is a father should remain a father in case of inconsistency, whereas an individual who has siblings should change its status and not be an only child anymore. Similarly for $\mathsf{a}\mathcal{T}_2$, where '$-\mathsf{hasSibling}.\top$' removes all relations between individuals that violate the axiom and individuals satisfying $\top$ (i.e., all individuals), thus

stating that an only child who has siblings should drop its 'sibling' relationship with everyone and stay an only child. Now consider the following ABox:

$$\mathcal{A} = \{\mathsf{John} : \mathsf{Male} \sqcap \mathsf{Father} \sqcap \neg\mathsf{Parent}, \mathsf{Mary} : \mathsf{OnlyChild}, \mathsf{hasSibling}(\mathsf{Mary}, \mathsf{John})\}$$

A repaired ABox then according to $\mathsf{a}\mathcal{T}_1$ should be the following:

$$\mathcal{A}_1 = \{\mathsf{John} : \mathsf{Male} \sqcap \mathsf{Father} \sqcap \mathsf{Parent}, \mathsf{Mary} : \neg\mathsf{OnlyChild}, \mathsf{hasSibling}(\mathsf{Mary}, \mathsf{John})\}$$

whereas a repaired ABox according to $\mathsf{a}\mathcal{T}_2$ should be the following:

$$\mathcal{A}_2 = \{\mathsf{John} : \mathsf{Male} \sqcap \neg\mathsf{Father} \sqcap \neg\mathsf{Parent}, \mathsf{Mary} : \mathsf{OnlyChild}\}$$

4 Contributions

We have attempted to materialize the above intuitions in a number of different formats and examine how the various definitions of repairs from the database literature translate and behave in the DL setting. This has resulted in several publications as well as the PhD thesis [22].

Our main goal was to apply the idea behind active integrity constraints in the Description Logic setting and, in particular, to provide extensions of TBox axioms so that they are able to suggest *preferred* repairing routes in case of inconsistencies. We started by overviewing active integrity constraints in the propositional setting through the DL-PA lens. Embeddings of *weak*, *founded* and *justified* repairs into DL-PA were proposed and studied in [13, 15]. The DL-PA framework moreover naturally leads to a a new, dynamic way of integrity maintenance, *dynamic repairs*. We have analysed the properties of all these repairs and provides complexity results for the problem of existence of dynamic repairs. We also took advantage of the dynamic framework—i.e., the logic DL-PA—in order to explore an extension on databases with history and adjust the behavior of the various repairs so that they work in this setting. Finally, we provided DL-PA counterparts of all the reasoning and decision problems arising in the repair setting, such as the existence of a repair or the existence of a unique repair.

In the paper [23] we have started to lift the idea behind $\mathcal{AIC}$s to Description Logics. There, the first definition of '*active*' TBoxes was introduced. After a brief discussion on the differences and difficulties of leaving the propositional setting, we examine preliminary steps into repairing ABoxes *syntactically* so that they conform to the preferences denoted by the active axioms. These syntactic repairs are inspired by (and correspond to) the weak and founded repairs of the database literature. Proving to be quite impractical though, we suggest that a *semantic* approach seems more viable.

In the paper [14] we went on to pursue this semantic approach and investigated how the dynamic logic-based framework can be transferred to the DL level. We use a more *local* approach, where preferred update actions in the active axioms behave similarly to the update actions introduced before, i.e., they

have the form $\pm A$ for an *atomic* concept A denoting either the *addition* or the *removal* of an individual from the set of individuals that have property A. This approach, although more expressive and well-behaved than the syntactic one of [23], still leaves a lot of repairing scenarios unattainable, mainly because of its boolean nature and close similarity to the repairs of the database literature.

In Rantsoudis's Phd thesis a more elaborate logic and techniques are introduced and discussed which apply changes *globally*, in the sense that preferred update actions of the form $\pm C$ can be applied to *all* individuals violating an axiom and C is not necessarily atomic [22, Chapter 5]. As before, a *dynamic logic* framework that is influenced by the logic DL-PA is used. Furthermore, although the resulting logics are extensions of $\mathcal{ALCO}$ and $\mathcal{ALCIO}$ respectively with dynamic operators, they are shown to be as expressive as their static counterparts (with the addition of the universal role).

The Phd thesis [22, Chapter 6] concludes with a proposal to connect the approaches of the syntactic approach in [23] and the semantic approach in [14], thus completing the picture on active TBox-based ABox repairs. It also briefly discusses future work, with possible applications of the proposed repairing methods on nonmonotonic scenarios.

References

1. Abiteboul, S.: Updates, A new frontier. In: Gyssens, M., Paredaens, J., Gucht, D.V. (eds.) ICDT'88, 2nd International Conference on Database Theory, Bruges, Belgium, August 31 - September 2, 1988, Proceedings. Lecture Notes in Computer Science, vol. 326, pp. 1–18. Springer (1988). https://doi.org/10.1007/3-540-50171-1_1, https://doi.org/10.1007/3-540-50171-1_1
2. Baader, F., Suntisrivaraporn, B.: Debugging SNOMED CT using axiom pinpointing in the description logic EL+. In: Cornet, R., Spackman, K.A. (eds.) Proceedings of the Third International Conference on Knowledge Representation in Medicine, Phoenix, Arizona, USA, May 31st - June 2nd, 2008. CEUR Workshop Proceedings, vol. 410. CEUR-WS.org (2008), http://ceur-ws.org/Vol-410/Paper01.pdf
3. Bertossi, L.E.: Database Repairing and Consistent Query Answering. Synthesis Lectures on Data Management, Morgan & Claypool Publishers (2011). https://doi.org/10.2200/S00379ED1V01Y201108DTM020, https://doi.org/10.2200/S00379ED1V01Y201108DTM020
4. Bienvenu, M., Bourgaux, C., Goasdoué, F.: Query-driven repairing of inconsistent dl-lite knowledge bases. In: Kambhampati, S. (ed.) Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016. pp. 957–964. IJCAI/AAAI Press (2016), http://www.ijcai.org/Abstract/16/140
5. Bienvenu, M., Bourgaux, C., Goasdoué, F.: Computing and explaining query answers over inconsistent dl-lite knowledge bases. J. Artif. Intell. Res. **64**, 563–644 (2019). https://doi.org/10.1613/jair.1.11395, https://doi.org/10.1613/jair.1.11395
6. Caroprese, L., Greco, S., Sirangelo, C., Zumpano, E.: Declarative semantics of production rules for integrity maintenance. In: Etalle, S., Truszczynski, M. (eds.) Logic Programming, 22nd International Conference, ICLP 2006, Seattle,

WA, USA, August 17-20, 2006, Proceedings. Lecture Notes in Computer Science, vol. 4079, pp. 26–40. Springer (2006). https://doi.org/10.1007/11799573_5, https://doi.org/10.1007/11799573_5
7. Caroprese, L., Greco, S., Zumpano, E.: Active integrity constraints for database consistency maintenance. IEEE Trans. Knowl. Data Eng. **21**(7), 1042–1058 (2009). https://doi.org/10.1109/TKDE.2008.226, https://doi.org/10.1109/TKDE.2008.226
8. Caroprese, L., Truszczynski, M.: Active integrity constraints and revision programming. TPLP **11**(6), 905–952 (2011). https://doi.org/10.1017/S1471068410000475, https://doi.org/10.1017/S1471068410000475
9. Ceri, S., Fraternali, P., Paraboschi, S., Tanca, L.: Automatic generation of production rules for integrity maintenance. ACM Trans. Database Syst. **19**(3), 367–422 (1994). https://doi.org/10.1145/185827.185828, http://doi.acm.org/10.1145/185827.185828
10. Chomicki, J., Marcinkowski, J.: Minimal-change integrity maintenance using tuple deletions. Inf. Comput. **197**(1-2), 90–121 (2005). https://doi.org/10.1016/j.ic.2004.04.007, https://doi.org/10.1016/j.ic.2004.04.007
11. Cruz-Filipe, L.: Optimizing computation of repairs from active integrity constraints. In: Beierle, C., Meghini, C. (eds.) Foundations of Information and Knowledge Systems - 8th International Symposium, FoIKS 2014, Bordeaux, France, March 3-7, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8367, pp. 361–380. Springer (2014). https://doi.org/10.1007/978-3-319-04939-7_18, https://doi.org/10.1007/978-3-319-04939-7_18
12. Cruz-Filipe, L., Gaspar, G., Engrácia, P., Nunes, I.: Computing repairs from active integrity constraints. In: Seventh International Symposium on Theoretical Aspects of Software Engineering, TASE 2013, 1-3 July 2013, Birmingham, UK. pp. 183–190. IEEE Computer Society (2013). https://doi.org/10.1109/TASE.2013.32, https://doi.org/10.1109/TASE.2013.32
13. Feuillade, G., Herzig, A.: A dynamic view of active integrity constraints. In: Fermé, E., Leite, J. (eds.) Logics in Artificial Intelligence - 14th European Conference, JELIA 2014, Funchal, Madeira, Portugal, September 24-26, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8761, pp. 486–499. Springer (2014). https://doi.org/10.1007/978-3-319-11558-0_34, https://doi.org/10.1007/978-3-319-11558-0_34
14. Feuillade, G., Herzig, A., Rantsoudis, C.: A dynamic extension of ALCO for repairing via preferred updates. In: Proceedings of the 31th International Workshop on Description Logics, Tempe, Arizona, October 27-29, 2018 (2018), to appear
15. Feuillade, G., Herzig, A., Rantsoudis, C.: A dynamic logic account of active integrity constraints. Fundamenta Informaticae (2019), to appear
16. Flesca, S., Greco, S., Zumpano, E.: Active integrity constraints. In: Moggi, E., Warren, D.S. (eds.) Proceedings of the 6th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming, 24-26 August 2004, Verona, Italy. pp. 98–107. ACM (2004). https://doi.org/10.1145/1013963.1013977, http://doi.acm.org/10.1145/1013963.1013977
17. Herzig, A., Rifi, O.: Propositional belief base update and minimal change. Artif. Intell. **115**(1), 107–138 (1999). https://doi.org/10.1016/S0004-3702(99)00072-7, https://doi.org/10.1016/S0004-3702(99)00072-7
18. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all justifications of OWL DL entailments. In: Aberer, K., Choi, K., Noy, N.F., Allemang, D., Lee, K., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber,

G., Cudré-Mauroux, P. (eds.) The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007. Lecture Notes in Computer Science, vol. 4825, pp. 267–280. Springer (2007). https://doi.org/10.1007/978-3-540-76298-0_20, https://doi.org/10.1007/978-3-540-76298-0_20

19. Meyer, T.A., Lee, K., Booth, R., Pan, J.Z.: Finding maximally satisfiable terminologies for the description logic ALC. In: Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA. pp. 269–274. AAAI Press (2006), http://www.aaai.org/Library/AAAI/2006/aaai06-043.php
20. Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between OWL and relational databases. J. Web Sem. **7**(2), 74–89 (2009). https://doi.org/10.1016/j.websem.2009.02.001, https://doi.org/10.1016/j.websem.2009.02.001
21. Parsia, B., Sirin, E., Kalyanpur, A.: Debugging OWL ontologies. In: Ellis, A., Hagino, T. (eds.) Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005. pp. 633–640. ACM (2005). https://doi.org/10.1145/1060745.1060837, http://doi.acm.org/10.1145/1060745.1060837
22. Rantsoudis, C.: Knowledge Bases and Preferred Update Actions : Searching for Consistency through Dynamic Logic Programs. (Bases de connaissance et actions de mise à jour préférées : à la recherche de consistance au travers des programmes de la logique dynamique). Ph.D. thesis, Paul Sabatier University, Toulouse, France (2018), https://tel.archives-ouvertes.fr/tel-02479121
23. Rantsoudis, C., Feuillade, G., Herzig, A.: Repairing ABoxes through active integrity constraints. In: Artale, A., Glimm, B., Kontchakov, R. (eds.) Proceedings of the 30th International Workshop on Description Logics, Montpellier, France, July 18-21, 2017. CEUR Workshop Proceedings, vol. 1879. CEUR-WS.org (2017), http://ceur-ws.org/Vol-1879/paper41.pdf
24. Reiter, R.: A theory of diagnosis from first principles. Artif. Intell. **32**(1), 57–95 (1987). https://doi.org/10.1016/0004-3702(87)90062-2, https://doi.org/10.1016/0004-3702(87)90062-2
25. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Gottlob, G., Walsh, T. (eds.) IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003. pp. 355–362. Morgan Kaufmann (2003), http://ijcai.org/Proceedings/03/Papers/053.pdf
26. Schlobach, S., Huang, Z., Cornet, R., van Harmelen, F.: Debugging incoherent terminologies. J. Autom. Reasoning **39**(3), 317–349 (2007). https://doi.org/10.1007/s10817-007-9076-z, https://doi.org/10.1007/s10817-007-9076-z
27. Suntisrivaraporn, B., Qi, G., Ji, Q., Haase, P.: A modularization-based approach to finding all justifications for OWL DL entailments. In: Domingue, J., Anutariya, C. (eds.) The Semantic Web, 3rd Asian Semantic Web Conference, ASWC 2008, Bangkok, Thailand, December 8-11, 2008. Proceedings. Lecture Notes in Computer Science, vol. 5367, pp. 1–15. Springer (2008). https://doi.org/10.1007/978-3-540-89704-0_1, https://doi.org/10.1007/978-3-540-89704-0_1
28. Tao, J., Sirin, E., Bao, J., McGuinness, D.L.: Integrity constraints in OWL. In: Fox, M., Poole, D. (eds.) Proceedings of the Twenty-Fourth AAAI Conference on Artifi-

cial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010. AAAI Press (2010), http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1931
29. Troquard, N., Confalonieri, R., Galliani, P., Peñaloza, R., Porello, D., Kutz, O.: Repairing ontologies via axiom weakening. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018. AAAI Press (2018), https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17189
30. Winslett, M.A.: Updating Logical Databases. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press (1990)